UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 09/930,598 | 08/15/2001 | Andrew James Osborne | GB920000082US1 | 1631 |

7590 05/11/2005

IBM Corp, IP Law Dept T81/503
3039 Cornwallis Road
PO Box 12195
Research Triangle Park, NC 27709-2195

| EXAMINER | |
|---|---|
| CHOW, CHIH CHING | |
| **ART UNIT** | **PAPER NUMBER** |
| 2192 | |

DATE MAILED: 05/11/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

| Office Action Summary | Application No. | Applicant(s) | |
|---|---|---|---|
| | 09/930,598 | OSBORNE, ANDREW JAMES | |
| | Examiner | Art Unit | |
| | Chih-Ching Chow | 2192 | |

*-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --*

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

1)☒ Responsive to communication(s) filed on *15 August 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

4)☒ Claim(s) *1-37* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-37* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

**Application Papers**

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *15 August 2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *12/10/01, 09/10/01*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

## DETAILED ACTION

1. This action is responsive to the application filed on August 15, 2001.

2. The priority date considered for this application is August 3, 2000, which is the filing date of the international patent application GB no. 0021148.2.

3. Claims 1-37 have been examined.

### *Claim Rejections - 35 USC § 101*

4. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

5. Claims 1-11 are rejected under 35 USC § 101 because the claimed invention is directed to non-statutory subject matter.

Statutory subject matter requires two things:

(1) it must be in the "useful arts," U.S. Const., art. I, § 8, cl. 8, which is equivalent to the modern "industrial" or "technological arts," defined by Congress in the four categories of "process, machine, manufacture, or composition of matter" in 35 USC § 101; and if it is,

(2) it must not fall within one of the exceptions for "laws of nature, physical phenomena and abstract ideas."

Under the most recent Federal Circuit cases, transformation of data by a machine (e.g., computer) is statutory subject matter provided the claims recite a "practical application, which produce[s] a useful, concrete and tangible result." State St. Bank & Trust CO. v. Signature Financial Group, Inc., 149 F. 3d 1368, 1373, 47 USPQ2d, 1596, 1600-01 (Fed. Cir. 1998).

Claim 1 recites:

'A method of validating a syntactical statement employing a stored syntax tree representing all possible syntax options by means of a network of junction nodes and data nodes between a root node and an end node, such that all paths through the tree lead to the end node, said method comprising the steps of:
passing said syntactical statement to the root node and parsing said syntactical statement into elementary tokens in the root node;
current node in the syntax tree, whereby said current node is initially the root node;
maintaining the location of a returning potential nodes that can be selected from the current node and their distances from the current
node;
returning potential nodes that can be selected from the current noted and their distances from the current node;
in response to said returning step, comparing the potential node to the stored tokens and selecting a potential node corresponding to one of said stored tokens if such a corresponding node exists;
updating the location of the current node to that of a selected node,
repeating said returning, comparing and selecting steps wherein the syntactical statement is validated if the end node is reached.'

In this instance, the language of the claim raises a question as to whether the claim is directed merely to an abstract idea that is not tied to a technological art, environment or machine which would result in a practical application producing a useful, concrete and tangible result to form the basis of statutory subject matter under 35 USC § 101.

Furthermore, the Office's interpretation of this claim is that it does not expressly or implicitly require performance of any of the steps by a machine such as a general-purpose digital computer. Structure will not be read into the claims for the purpose of the statutory subject matter analysis even though the steps might be capable of being performed by a machine.

On this basis, claim 1 is rejected under 35 USC § 101 as being directed to nonstatutory subject matter. Claims 2-11, which depend from claim 1, are all rejected under 35 USC § 101 for the same reasons.

## *Claim Rejections - 35 USC § 102*

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless -
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

7. Claims 1, 4, 6-9, 11-12, 15, 18-21, 23-27, 30 and 37 are rejected under 35 U.S.C. 102(b) as being anticipated by "Compilers, Principles, Techniques, and Tools", by Alfred V. Aho et al. (hereinafter "Aho").

| **CLAIM** | **Aho** |
|---|---|
| 1. A method of validating a syntactical statement employing a stored syntax tree representing all possible syntax options by means of a network of junction nodes and data nodes between a root node and an end node, such that all paths through the tree lead to the end node, said method comprising the steps of:<br>a. passing said syntactical statement to the root node and parsing said syntactical statement into elementary tokens in the root node;<br>b. current node in the syntax tree, | Aho teaches syntactical analysis, parse tree, tokens, etc in his book. See page 11, "A **symbol table** is a data structure containing a record for identifiers, e.g., the names of variables, arrays, and functions, and the fields and attributes of identifiers"; page 29, "Parse Tree"; page 60, section 2.7 "INCORPORATING A SYMBOL TABLE" (*passing syntactical statement for parsing, and put token into symbol table*); page 98, section 3.4 "RECOGNITION OF TOKENS" (*syntactical statement into elementary tokens*); pages 99-101, 114, 115 and |

whereby said current node is initially the root node;

c. maintaining the location of a returning potential nodes that can be selected from the current node and their distances from the current node;

d. returning potential nodes that can be selected from the current noted and their distances from the current node;

e. in response to said returning step, comparing the potential node to the stored tokens and selecting a potential node corresponding to one of said stored tokens if such a corresponding node exists;

f. updating the location of the current node to that of a selected node,

g. repeating said returning, comparing and selecting steps wherein the syntactical statement is validated if the end node is reached.

"Transition Diagrams", a 'transition state diagram' has nodes at different locations of the string (*maintaining the location of a returning potential nodes*), it starts at a **start state**, which is the 'root node'; it ends at a **final state**. There are 'accepting states' (page 100) which joins different states together (junction node). When the parsing moves along with the syntactic string, the state (node) moves forward, and the symbol table also gets updated (page 115). The entire process repeats till it hits the end of the string.

4. A method as claimed in claim 1, in which the distance between a potential node and said current node is measured by enumerating the number of nodes between said potential node and said current node.

For the feature of claim 1 see claim 1 rejection. Aho teaches measuring distance in his book, see page 155, under 3.34, "Define d(x, y), the **distance** between x and y, to be the minimum number of insertions and deletions required to transform x into y."

6. A method as claimed in claim 1, in which said syntax tree comprises branched nodes, whereby said branched nodes represent optional tokens or a

For the feature of claim 1 see claim 1 rejection. Aho teaches a branched transition, see Aho page 101, Fig. 3.12, the branch of the state diagram is a

| | |
|---|---|
| start node of a sub-tree. | 'sub-tree', which starts with an optional token or a start node of a sub-tree, and followed with a junction node (accepting state) and/or data nodes. The sub-state can have another sub-state nested in it, see Fig. 3.12. |
| 7. A method as claimed claim 6, in which if a branched node represents a start node of a sub-tree, said sub-tree comprises further junction nodes and/or data nodes. | Same as claim 6 rejection. |
| 8. A method as claimed in claim 7, in which sub-trees are nested hierarchically if a sub-tree comprises at least one further start node of a sub-tree. | Same as claim 6 rejection. |
| 9. A method as claimed in claim 6, which said comparing step further includes the step of:<br>verifying if a potential node is a start node of a sub-tree. | Same as claim 6 rejection. |
| 11. A method as claimed in claim 1, in which said syntactical statement comprises a textual string. | For the feature of claim 1 see claim 1 rejection. Aho's teaching is for parsing a textual string. See Aho page 98-104. |
| 12. A system for validating a syntactical statement comprising:<br>a. a stored syntax tree representing possible syntax options by means of a network of junction nodes and data nodes between a root node and an end node, such that all paths through the | Same as claim 1 rejection; Aho's teaching is for a system to validating a syntactical statement. |

tree lead to the end node, whereby said syntactical statement is initially passed to the root node;

b. means for parsing said syntactical statement into elementary tokens in the root node;

c. a table to store the tokens, and entries representing the end node of the syntactical statement;

d. means for maintaining the location of a current node in the syntax tree, whereby said current node is initially the root node;

e. means for returning potential nodes that can be selected from the current node and their distances from the current node;

f. means for comparing the potential nodes to the stored tokens and means for selecting a potential node of said potential nodes corresponding to one of said stored tokens if such a corresponding node exists; and

g. means for updating the location of the current node;

h. whereby said syntactical statement is valid if said end node is reached.

15. A system as claimed in claim 12, further comprising:

means for enumerating the number of nodes between said potential node and said current node, in order to provide the distance between a potential node and said current node.

For the feature of claim 12 see claim 12 rejection, for the rest of claim 15 feature see claim 4 rejection.

| | |
|---|---|
| 18. A system as claimed in claim 12, in which said syntax tree further comprises:<br>branched nodes, whereby said branched nodes represent optional tokens or a start node of a sub-tree. | For the feature of claim 12 see claim 12 rejection, for the rest of claim 18 feature see claim 6 rejection. |
| 19. A system as claimed in claim 18, in which if a branched node represents a start node of a sub-tree, said sub-tree comprises:<br>further junction nodes and/or data nodes. | For the feature of claim 18 see claim 18 rejection, for the rest of claim 19 feature see claim 7 rejection. |
| 20. A system as claimed in claim 19, in which said system further comprises:<br>means for nesting sub-trees hierarchically if a sub-tree comprises at least one further start node of a sub-tree. | For the feature of claim 19 see claim 19 rejection, for the rest of claim 20 feature see claim 8 rejection. |
| 21. A system as claimed in claim 20, in which said comparing means further comprises:<br>means for verifying successfully if a potential node is a start node of a sub-tree. | For the feature of claim 20 see claim 20 rejection, for the rest of claim 21 feature see claim 9 rejection. |
| 23. A system as claimed in claim 12, in which said syntactical statement is a textual string. | For the feature of claim 12 see claim 12 rejection, for the rest of claim 23 feature see claim 11 rejection. |
| 24. A syntax checker comprising:<br>a stored syntax tree representing a body of valid syntax; and<br>a table for holding elementary tokens | Aho's teaching is actually more than a syntax checker. See Claim 1 rejection. |

of syntax making up a syntactical statement to be checked;

said syntax tree comprising root and end node objects joined by a network of junction node objects and data node objects, said data node objects representing options in the syntax including tokens, such that each junction node object may link to an unlimited number of other junction node objects and data node objects, and each data node object only links to a singular junction node object, whereby all pathways through the network eventually terminate in said end node object;

each of said junction node objects, being effective to evaluate linked data node options following said junction node object so that any tokens in said table corresponding to a linked data node object are marked as "found",

whereby said syntactical statement is progressively compared with said syntax tree either until said end node is reached, indicating the syntax of the statement is valid, or until a corresponding data object is not found, indicating said syntax is not valid.

25. A syntax checker as claimed in claim 24 wherein said syntax tree further comprises a sub-tree having a start node represented by a corresponding syntax option in a data node.

For the feature of claim 24 see claim 24 rejection, for the rest of claim 25 feature see claim 7 rejection.

| | |
|---|---|
| 26. A syntax checker as claimed in claim 24 wherein said syntax tree comprises a plurality of sub-trees, nested hierarchically, each nested sub-tree having a respective start node represented by a corresponding syntax option in a data node. | For the feature of claim 24 see claim 24 rejection, for the rest of claim 26 feature see claim 8 rejection. |
| 27. A computer program recorded on a medium consisting of instructions which, when executed on a computer, perform a method of validating a syntactical statement employing a stored syntax tree representing all possible syntax options by means of a network of junction nodes and data nodes between a root node and an end node, such that all paths through the tree lead to the end node, said method comprising the steps of:<br>passing said syntactical statement to the root node and parsing said syntactical statement into elementary tokens in the root node;<br>maintaining the location of a current node in the syntax tree, whereby said current node is initially the root node;<br>returning potential nodes that can be selected from the current node and their distances from the current node;<br>in response to said returning step, comparing the potential nodes to the stored tokens and selecting a potential node corresponding to one said stored tokens if such a corresponding node exists; | Same as claim 1 rejection above. |

updating the location of the current node to that of a selected node,

repeating said returning, comparing and selecting steps wherein the syntactical statement is validated if the end node is reached.

| | |
|---|---|
| 30. A computer program as claimed in claim 27, in which the distance between a potential node and said current node is measured by enumerating the number of nodes between said potential node and said current node. | For the feature of claim 27 see claim 27 rejection, for the rest of claim 30 feature see claim 4 rejection. |
| 37. A computer program as claimed in claim 27, in which said syntactical statement comprises a textual string. | For the feature of claim 27 see claim 27 rejection, for the rest of claim 37 feature see claim 11 rejection. |

## *Claim Rejections - 35 USC § 102*

8. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless -
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

9. Claims 1, 6-9, 12, 24, 27, 32-35 are rejected under 35 U.S.C. 102(e) as being anticipated by US Patent No. 6,378,126 by Min-Mei Tang (hereinafter "Tang").

The applied reference has a common assignee with the instant application. Based upon the earlier effective U.S. filing date of the reference, it constitutes prior art under 35 U.S.C. 102(e). This rejection under 35 U.S.C. 102(e) might be overcome either by a showing under 37 CFR 1.132 that any invention disclosed but not claimed in the reference was derived from the inventor of this application and is thus not the invention "by another," or by an appropriate showing under 37 CFR 1.131.

| CLAIM | Tang |
|---|---|
| 1. A method of validating a syntactical statement employing a stored syntax tree representing all possible syntax options by means of a network of junction nodes and data nodes between a root node and an end node, such that all paths through the tree lead to the end node, said method comprising the steps of:<br>a. passing said syntactical statement to the root node and parsing said syntactical statement into elementary tokens in the root node;<br>b. current node in the syntax tree, whereby said current node is initially the root node;<br>c. maintaining the location of a returning potential nodes that can be selected from the current node and their distances from the current node;<br>d. returning potential nodes that can be selected from the current noted and their distances from the current node; | See Tang column 1, lines 31-37, "**parse tree** phase follows where the **source statements** are converted into a **parse tree** which describes the **syntactic structure** of a source statement. A **parse tree** may be expressed as a **syntax tree** in which the operators appear as interior **nodes** and the operands of an operator are the children of the **node** for that operator" and see Tang's example on Fig. 3, and column 5, lines 37-40, "The **parse tree** 70 includes four **root nodes** 72a, b, c, d for each component API function call for the SQL statement." Fig. 3 shows the parse tree ends at an 'End' node. For item a, see Tang column 1, lines 25-28, "The compiler **processes the source code** in phases. In the first phase, the lexical scanning phase, the compiler groups the characters of a source program into **tokens**, which are logically cohesive sequences of characters." Also see Tang claim 25, "a **root node** for each |

e. in response to said returning step, comparing the potential node to the stored tokens and selecting a potential node corresponding to one of said stored tokens if such a corresponding node exists;

f. updating the location of the current node to that of a selected node,

g. repeating said returning, comparing and selecting steps wherein the syntactical statement is validated if the end node is reached.

function call for the program statement, wherein the program statement is executed (*current node*) by at least one function call and wherein the function calls are provided by an application precompiler that receives program statements from the compiler and generates and **returns** function calls capable of **executing the program statement**; (*maintaining the location of a returning potential nodes*)" See Tang column 1, lines 39-44, "the **parsed tree** may then be **optimized in manners** known in the art to develop the **shortest linked** (*keeping track of the 'distances' between the current node and the root node)* lists providing a structure of the code. Another phase of a compiler is the **generation of a symbol table**."

6. A method as claimed in claim 1, in which said syntax tree comprises branched nodes, whereby said branched nodes represent optional tokens or a start node of a sub-tree.

For the feature of claim 1 see claim 1 rejection. For the rest of claim 6 feature see Tang column 5, lines 40-43, "each root node has a pointer to a lower node for a **subcall** (sub-tree) and a **pointer horizontally to the next root node** (*junction nodes and/or data nodes*) forming a chain of the parse trees that comprise the SQL statement."

7. A method as claimed claim 6, in which if a branched node represents a start node of a sub-tree, said sub-tree comprises further junction nodes and/or data nodes.

Same as claim 6 rejection.

8. A method as claimed in claim 7, in which sub-trees are nested hierarchically if a sub-tree comprises at least one further start node of a sub-tree.

Same as claim 6 rejection.

9. A method as claimed in claim 6, which said comparing step further includes the step of: verifying if a potential node is a start node of a sub-tree.

Same as claim 6 rejection.

12. A system for validating a syntactical statement comprising:
a stored syntax tree representing possible syntax options by means of a network of junction nodes and data nodes between a root node and an end node, such that all paths through the tree lead to the end node, whereby said syntactical statement is initially passed to the root node;
means for parsing said syntactical statement into elementary tokens in the root node;
a table to store the tokens, and entries representing the end node of the syntactical statement;
means for maintaining the location of a current node in the syntax tree, whereby said current node is initially the root node;
means for returning potential nodes that can be selected from the current node and their distances from the current node;
means for comparing the potential

Same as claim 1 rejection above; Tang's teaching is for a system to validating a syntactical statement; see Tang's Abstract, "Disclosed is a system and method for compiling a program."

nodes to the stored tokens and means for selecting a potential node of said potential nodes corresponding to one of said stored tokens if such a corresponding node exists; and

means for updating the location of the current node;

whereby said syntactical statement is valid if said end node is reached.

24. A syntax checker comprising:

a stored syntax tree representing a body of valid syntax; and

a table for holding elementary tokens of syntax making up a syntactical statement to be checked;

said syntax tree comprising root and end node objects joined by a network of junction node objects and data node objects, said data node objects representing options in the syntax including tokens, such that each junction node object may link to an unlimited number of other junction node objects and data node objects, and each data node object only links to a singular junction node object, whereby all pathways through the network eventually terminate in said end node object;

each of said junction node objects, being effective to evaluate linked data node options following said junction node object so that any tokens in said table corresponding to a linked data node object are marked as "found",

Tang's disclosure is also a syntax checker, see Tang column 1, lines 30-34, "during a parsing phase, the **syntax** and semantics of the tokens are **checked for errors**. A parse tree phase follows where the source statements are converted into a parse tree which describes the **syntactic structure** of a source statement." For the rest of claim 24 feature see Tang's claim 1 rejection.

whereby said syntactical statement is progressively compared with said syntax tree either until said end node is reached, indicating the syntax of the statement is valid, or until a corresponding data object is not found, indicating said syntax is not valid.

27. A computer program recorded on a medium consisting of instructions which, when executed on a computer, perform a method of validating a syntactical statement employing a stored syntax tree representing all possible syntax options by means of a network of junction nodes and data nodes between a root node and an end node, such that all paths through the tree lead to the end node, said method comprising the steps of:

Same as claim 1 rejection above.

passing said syntactical statement to the root node and parsing said syntactical statement into elementary tokens in the root node;

maintaining the location of a current node in the syntax tree, whereby said current node is initially the root node;

returning potential nodes that can be selected from the current node and their distances from the current node;

in response to said returning step, comparing the potential nodes to the stored tokens and selecting a potential node corresponding to one said stored tokens if such a corresponding node exists;

updating the location of the current node to that of a selected node,

repeating said returning, comparing and selecting steps wherein the syntactical statement is validated if the end node is reached.

| | |
|---|---|
| 32. A computer program as claimed in claim 27, in which said syntax tree comprises branched nodes, whereby said branched nodes represent optional tokens or a start node of a sub-tree. | For the feature of claim 27 see claim 27 rejection, for the rest of claim 32 feature see claim 6 rejections. |
| 33. A computer program as claimed in claim 32, in which if a branched node represents a start node of a sub-tree, said sub-tree comprises further junction nodes and/or data nodes. | For the feature of claim 32 see claim 32 rejection, for the rest of claim 33 feature see claim 7 rejections. |
| 34. A computer program as claimed in claim 32, in which sub-trees are nested hierarchically if a sub-tree comprises at least one further start node of a sub-tree. | For the feature of claim 32 see claim 32 rejection, for the rest of claim 34 feature see claim 8 rejections. |
| 35. A computer program as claimed in claim 32, in which said method comparing step further includes step of: verifying if a potential node is a start node of a sub-tree. | For the feature of claim 32 see claim 32 rejection, for the rest of claim 35 feature see claim 9 rejections. |

## *Claim Rejections - 35 USC § 103*

10. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

11. Claims 2-3, 5, 13-14, 16-17, 28-29, and 31 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Compilers, Principles, Techniques, and Tools", by Alfred V. Aho et al. (hereinafter "Aho"), in view of U.S. Patent No. 5,325,531 by William M. McKeeman et al (hereinafter "McKeeman").

| CLAIM | Aho / McKeeman |
|---|---|
| 2. A method as claimed the step of: claim 1, further comprising the step of: creating a table to store said tokens and an entry representing said end node and marking said tokens as "found" in response to said selection of a potential node. | For the feature of claim 1 see claim 1 rejection. Aho's book teaches fundamentals of parsing, syntax trees, symbol table, tokens, and transition diagrams etc. but he does not mention 'marking said tokens in the token table' specifically, however, McKeeman teaches it in an analogous prior art. In McKeeman's FIGURE 7a, FIGURE 8 (see column 5, line 5, ",FIG. 8 is a diagram of the structure of a **token table** generated in the compiler of FIG. 7); a token table is created for each of the tokens, and an entry representing the tokens is in a symbol table entry. The attribute for that entry can specify any information related to that token, such as 'found' or 'not found', see McKeeman's column 15, lines 10-20, "the journalled actions are scope entry and exit, symbol lookup and enter, and **get and set for any attribute**. Typically the symbol enter and attribute setting is done in |

| | |
|---|---|
| | response to a declarative construct in the application language. Typically the symbol lookup and attribute getting is done in response to an executable construct in the application language. Atypically there are situations which can cause any of the actions in association with any of the application language constructs"<br>It would have been obvious to a person of ordinary skill in the art at the time of the invention was made to supplement Aho's disclosure of the compiling fundamentals by using marking the symbol table taught by McKeeman, for keeping track of the lexical unit information (McKeeman column 13, lines 57-60). |
| 3. A method as claimed in claim 2, comprising the further step of:<br>initially marking said tokens and end node entry in the table as "not found". | For the feature of claim 2 see claim 2 rejection, same as claim 2 rejection. |
| 5. A method as claimed in claim 2, in which said step of selecting a potential node further comprises the steps of:<br>a. verifying successfully if said potential node corresponds to a stored token;<br>b. verifying successfully if said potential node is closest in distance to said current node compared with the remaining potential nodes, and<br>c. marking said stored token as "found" in the table. | For the feature of claim 2 see claim 2 rejection. For verifying the node, see McKeeman column 25, lines 29-30, "the contents of the semantic increment in the semantic increment table 76 are checks for **validating** it (*verifying*) and a handle into a code increment table 73." the marking a "found", can be part of the attributes in the table. |

| | |
|---|---|
| 13. A system as claimed in claim 12, in which said system further comprises:<br>a. means for marking said tokens as "found" in response to selection of a potential node by said selecting means. | For the feature of claim 12 see claim 12 rejection, for the rest of claim 13 feature see claim 2 rejection. |
| 14. A system as claimed in claim 12 further including means for initially marking said tokens and end node entries in the table as "not found". | For the feature of claim 12 see claim 12 rejection, for the rest of claim 14 feature see claim 2 rejection. |
| 16. A system as claimed in claim 12, in which said means for selecting a potential node further comprises:<br>first means for verifying if said potential node corresponds to a stored token; and<br>second means for verifying if said potential node is closest in distance to said current node compared with the remaining potential nodes. | For the feature of claim 12 see claim 12 rejection, for the rest of claim 16 feature see claim 5 rejection. |
| 17. A system as claimed in claim 12, in which said system further comprises:<br>if the end node is reached, means for confirming the syntactical statement is valid if all stored tokens, including the end node entry in the table are marked as "found". | For the feature of claim 12 see claim 12 rejection, for the rest of claim 17 feature see claim 2, 3 and 5 rejections. |
| 28. A computer program as claimed in claim 27, in which said method further comprises the step of:<br>creating a table to store said tokens and an entry representing said end node | For the feature of claim 27 see claim 27 rejection, for the rest of claim 28 feature see claim 2 rejection. |

and marking said tokens as "found" in response to said selection of a potential node.

29. A computer program as claimed in claim 28, comprising the further method step of:
initially marking said tokens and end node entry in the table as "not found".

For the feature of claim 28 see claim 28 rejection, for the rest of claim 29 feature see claim 3 rejections.

31. A computer program as claimed in claim 27, in which said step of selecting a potential node further comprises the steps of:
verifying successfully if said potential node corresponds to a stored token;
verifying successfully if said potential node is closest in distance to said current node compared with the remaining potential nodes, and marking said stored token as "found" in the table.

For the feature of claim 27 see claim 27 rejection, for the rest of claim 31 feature see claim 5 rejections.

12. Claims 10, 22, and 36 are rejected under 35 U.S.C. 103(a) as being unpatentable over "Compilers, Principles, Techniques, and Tools", by Alfred V. Aho et al. (hereinafter "Aho"), in view of U.S. Patent No. 5,546,507 by Wendy Staub (hereinafter "Staub").

| CLAIM | Aho / Staub |
| --- | --- |
| 10. A method as claimed in claim 1, in which said junction nodes are linked to any number of junction nodes or data nodes and in which said data nodes are linked to a single junction node. | For the feature of claim 4 see claim 4 rejection. Aho teaches all aspects of claim 10, but he does not mention 'junction node' (*connection node)* specifically, however, Staub teaches it in an analogous prior art. In Staub's, column 12, lines 13-15, "The system of the present invention checks all definitions employed in a decision **tree to validate** their correctness before attempting to generate code." And see Staub, column 5, lines 10-14, "As defined in the present invention, a node is a discrete point in a decision tree. A **node may represent a junction** where a decision is made, a possible goal in the decision process, or a **connection to another node** not included in the present decision tree. Each **node of the tree is connected to at least one other node** by a **path** leg"<br>It would have been obvious to a person of ordinary skill in the art at the time of the invention was made to supplement Aho's teaching of the compiling fundamentals by using junction nodes and data nodes taught by Staub, for the purpose of using graphical representation to create a logical tree (Staub column 1, lines 9-10). |
| 22. A system as claimed in claim 12, in which said system further comprises:<br>junction nodes that are linked to any number of junction nodes or data nodes | For the feature of claim 12 see claim 12 rejection. For the rest of the feature of claim 22 see claim 10 rejection. |

and data nodes that are linked to a single junction node.

36. A computer program as claimed in claim 27, in which said junction nodes are linked to any number of junction nodes or data nodes and in which said data nodes are linked to a single junction node.

For the feature of claim 27 see claim 27 rejection. For rest of the claim 36 feature see claim 10 rejection.

***Conclusion***

The following summarizes the status of the claims:

35 USC § 101 rejection: Claims 1-11

35 USC § 102 rejection: 1, 4, 6-9, 11-12, 15, 18-21, 23-27, 30, 32-35, 37

35 USC § 103 rejection: 2-3, 5, 13-14, 16-17, 28-29, 31, and 36

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Chih-Ching Chow whose telephone number is 571-272-3693. The examiner can normally be reached on 7:30am - 4:00pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306. Any inquiry of a general nature of relating to the status of this application should be directed to the **TC2100 Group receptionist: 571-272-2100.** Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR.

Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Chih-Ching Chow

Examiner

Art Unit 2192

May 05, 2005

CC

ANTONY NGUYEN-BA
PRIMARY EXAMINER